Kinetic energy conserving integrators for Gaussian thermostatted SLLOD

Fei Zhang, Debra J. Searles, Denis J. Evans, Jan S. den Toom Hansen, and Dennis J. Isbister

Citation: The Journal of Chemical Physics **111**, 18 (1999); doi: 10.1063/1.479358

View online: http://dx.doi.org/10.1063/1.479358

View Table of Contents: http://scitation.aip.org/content/aip/journal/jcp/111/1?ver=pdfcov

Published by the AIP Publishing

Kinetic energy conserving integrators for Gaussian thermostatted SLLOD

Fei Zhang
Department of Computational Science, Faculty of Science, National University of Singapore, Singapore 119260 and Research School of Chemistry, Australian National University, ACT 0200, Australia

Debra J. Searles
Department of Chemistry, University of Queensland, Brisbane Qld 4072, Australia

Denis J. Evans
Research School of Chemistry, Australian National University, ACT 0200, Australia

Jan S. den Toom Hansen and Dennis J. Isbister
School of Physics, University of New South Wales, University College, ADFA, Canberra ACT 2600, Australia

(Received 20 October 1998; accepted 5 April 1999)

A new integration scheme is developed for nonequilibrium molecular dynamics simulations where the temperature is constrained by a Gaussian thermostat. The utility of the scheme is demonstrated by its application to the SLLOD algorithm which is the standard nonequilibrium molecular dynamics algorithm for studying shear flow. Unlike conventional integrators, the new integrators are constructed using operator-splitting techniques to ensure stability and that little or no drift in the kinetic energy occurs. Moreover, they require minimum computer memory and are straightforward to program. Numerical experiments show that the efficiency and stability of the new integrators compare favorably with conventional integrators such as the Runge–Kutta and Gear predictor–corrector methods. © *1999 American Institute of Physics.* [S0021-9606(99)50125-6]

I. INTRODUCTION

In recent years, symplectic numerical integrators have been extensively used to study systems under Hamiltonian dynamics because of their intrinsic stability.[1–8] This stability is a result of the fact that they are the exact solution of a perturbed Hamiltonian which differs from the original Hamiltonian: this difference depends on the timestep of the integrator and remains bounded.[2,9–11] Due to the enhanced stability of symplectic integrators, one is able to use larger timesteps than is possible for nonsymplectic integration schemes and furthermore, longer simulation times can be achieved without *ad hoc* rescaling of the energy. In molecular dynamics simulations of equilibrium fluids, the second-order Verlet integration schemes are symplectic. Higher order symplectic integrators suitable for the investigation of equilibrium systems have also been developed.[2–4,6,8]

Symplectic integrators conserve the Jacobian of the dynamics, which is unity. For nonHamiltonian systems, one approach that may be taken to enhance stability is to ensure that the integrator generates the correct Jacobian.[8] This requirement may lead to fluctuations in other conserved quantities, such as the energy, however a stable integrator will eliminate drift from these quantities.

Although initial studies of the properties of simple liquids were carried out in the microcanonical ensemble, studies of more complex liquids in other equilibrium statistical mechanics ensembles are possible by the use of constraints. Nosé–Hoover feedback mechanisms and Gaussian constraints[12] are deterministic schemes that are often used. In many cases, modification of the dynamics to incorporate these schemes results in equations of motion which are no longer Hamiltonian, and thus symplectic integrators are not applicable. Nevertheless, it is possible to construct symplectic-like integrators for these nonHamiltonian systems by using operator-splitting techniques. For example, stable integrators for Nosé–Hoover thermostatted equations can be obtained using operator-splitting techniques.[5,11,13]

The Gaussian thermostat multiplier allows the isokinetic ensemble to be studied by ensuring that the kinetic temperature is fixed at all times in the simulation. It removes heat by applying a force of constraint which is parallel to the peculiar velocity of each particle in the system. However, if the Gaussian thermostatted equations of motion are integrated using a standard numerical integrator, the temperature will drift due to numerical error. Recently, Zhang[14] has used operator splitting techniques to develop integrators that are highly stable and ensure that the temperature remains fixed at each timestep (although in this case, the exact Jacobian is not generated at each timestep). Numerical tests on this integrator demonstrate its accuracy and stability.[14]

In order to study systems which are not at equilibrium, nonequilibrium molecular dynamics algorithms have been developed. Again, many of these algorithms employ equations of motion that are nonHamiltonian and therefore symplectic numerical integrators are not applicable. Nonequilibrium dynamics are usually dissipative, and thermostats such as Nosé–Hoover and Gauss are often used to ensure that the required temperature is maintained. In general, the equations of motion used in these algorithms are not symplectic, even if the unthermostatted equations of motion are Hamiltonian. In this work we develop stable numerical integrators for nonequilibrium molecular dynamics (NEMD) when a Gaussian

thermostat is used. (For Nosé–Hoover thermostatted NEMD, it is relatively easy to construct operator-splitting integrators, see, e.g., Ref. 15–18.) As in the case of the Gauss thermostatted equilibrium system,[14] the integrators are designed to ensure that the kinetic temperature is conserved at each timestep; the exact Jacobian is only generated in the limit of infinitely small timestep. Numerical tests indicate that the proposed integrator is stable and accurate.

As an example, consider the Gauss thermostatted SLLOD algorithm for the simulation of planar Couette flow. The adiabatic and thermostatted equations of motion are nonHamiltonian[12] and the thermostatted equations are given by:

$$\dot{\mathbf{q}}_i = \mathbf{p}_i/m + \mathbf{i}\gamma q_{yi}, \tag{1}$$

$$\dot{\mathbf{p}}_i = \mathbf{F}_i - \mathbf{i}\gamma p_{yi} - \alpha_g \mathbf{p}_i, \tag{2}$$

$$\dot{d}_x(t) = \gamma, \tag{3}$$

where the position of particle i is $\mathbf{q}_i = (q_{xi}, q_{yi}, q_{zi})^T$, the momentum is $\mathbf{p}_i = (p_{xi}, p_{yi}, p_{zi})^T$, $\mathbf{i}$ is the unit vector in the x-direction, $\mathbf{F}_i$ is the force on the ith particle, d_x is the lattice strain associated with the Lees–Edwards periodic boundary conditions[12,19] and α_g is the Gaussian thermostat multiplier:

$$\alpha_g = \frac{1}{2mK_0}\sum_{j=1}^{N}[\mathbf{F}_j\cdot\mathbf{p}_j - \gamma p_{xj}p_{yj}],$$
$$K_0 = \frac{1}{2m}\sum_{j=1}^{N}\mathbf{p}_j(0)\cdot\mathbf{p}_j(0). \tag{4}$$

The Gaussian thermostat fixes the instantaneous kinetic energy along a trajectory, $K(t) = (1/2m)\Sigma_{j=1}^{N}\mathbf{p}_j\cdot\mathbf{p}_j = K_0$. When a conventional algorithm is used to numerically integrate such equations of motion, the kinetic energy will typically drift away from its initial value (see Refs. 20, 21 and the present paper). In order to prevent drift, one has to use an *ad hoc* rescaling of velocities, or use an additional feedback term in the equations of motion. However, such *ad hoc* methods may induce undesired and unknown perturbation effects on the simulation results if the timestep is not sufficiently small. The main objective of the present paper is to develop several kinetic energy conserving integrators for the Gaussian thermostatted SLLOD equations.[22] These integrators can be easily applied to some other nonHamiltonian NEMD algorithms such as the ''Evans'' heat-flow algorithm.[23] The new integrators are based on operator-splitting techniques. They are straightforward to program, and require much less computer memory than some conventional algorithms. Our numerical tests indicate that the new integrators are very stable and efficient, and compare very favorably with the Runge–Kutta methods and Gear predictor–corrector methods that are widely used in NEMD simulations.

II. OPERATOR-SPLITTING INTEGRATORS

Recently operator-splitting techniques[1,2] have been used to develop efficient numerical integrators for molecular dynamics simulations.[5,11,13–15,17,18] The trajectory obtained from the solution of the equations of motion can be represented as the action of a propagator, $U(t)$, on the initial phase, $\mathbf{\Gamma}(0) = (\mathbf{q},\mathbf{p})^T$:

$$\mathbf{\Gamma}(t) = U(t)\mathbf{\Gamma}(0) = \exp(\mathbf{iL}t)\mathbf{\Gamma}(0), \tag{5}$$

where $\mathbf{iL}$ is the phase variable Liouvillean,

$$\mathbf{iL} = \dot{\mathbf{\Gamma}}\cdot\frac{\partial}{\partial\mathbf{\Gamma}}. \tag{6}$$

If we consider a discrete time propagation, then the time interval is divided into timesteps, of length Δt, and the propagator acts at each timestep:

$$\mathbf{\Gamma}(t+\Delta t) = U(\Delta t)\mathbf{\Gamma}(t) = \exp\left(\Delta t\dot{\mathbf{\Gamma}}\cdot\frac{\partial}{\partial\mathbf{\Gamma}}\right)\mathbf{\Gamma}(t). \tag{7}$$

The main idea of the operator-splitting methods is to replace the propagator, $U(\Delta t)$, by a combination of operators which approximate $U(\Delta t)$ to a known degree of accuracy. This is equivalent to splitting the coupled first-order differential equations under consideration [such as Eqs. (1), (2), and (3)] into two or more first-order differential equations, each of which can be solved either exactly or approximately. The Lie–Trotter formula approximates $U(\Delta t)$ by:

$$\begin{aligned} U(\Delta(t)) &= \exp(\mathbf{iL}\Delta t) \\ &= \exp((\mathbf{iL}_1 + \mathbf{iL}_2)\Delta t) \\ &= \prod_{i=1}^{n}\exp(a_i\mathbf{iL}_1\Delta t)\exp(b_i\mathbf{iL}_2\Delta t) + O(\Delta t^{r+1}), \end{aligned} \tag{8}$$

where a_i and b_i are parameters that are determined so that Eq. (8) is correct to terms in Δt^r. See Refs. 1, 2, 5, 11, 13–15, 17, 18 for more details.

For the Gaussian thermostatted SLLOD Eqs. (1), (2), and (3), the discrete time propagator is given by

$$U(\Delta t) = \exp\left(\Delta t\sum_{i=1}^{N}\begin{pmatrix}\mathbf{p}_i/m + \mathbf{i}\gamma q_{yi}\\ \mathbf{F}_i - \mathbf{i}\gamma p_{yi} - \alpha_g\mathbf{p}_i\\ \gamma\end{pmatrix}\cdot\begin{pmatrix}\partial/\partial\mathbf{q}_i\\ \partial/\partial\mathbf{p}_i\\ \partial/\partial d_x\end{pmatrix}\right). \tag{9}$$

We decompose the right-hand-side of Eq. (9) into two parts:

$$\begin{aligned} U(\Delta t) = \exp\Bigg(\Delta t\sum_{i=1}^{N}\Bigg(&\begin{pmatrix}\mathbf{p}_i/m + \mathbf{i}\gamma q_{yi}\\ \mathbf{0}\\ \gamma\end{pmatrix} \\ &+ \begin{pmatrix}\mathbf{0}\\ \mathbf{F}_i - \mathbf{i}\gamma p_{yi} - \alpha_g\mathbf{p}_i\\ 0\end{pmatrix}\Bigg)\cdot\begin{pmatrix}\partial/\partial\mathbf{q}_i\\ \partial/\partial\mathbf{p}_i\\ \partial/\partial d_x\end{pmatrix}\Bigg) \\ \equiv \exp\left(\Delta t(\mathbf{A}+\mathbf{B})\cdot\frac{\partial}{\partial\mathbf{\Gamma}}\right), \end{aligned} \tag{10}$$

where $\mathbf{A} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_2, \ldots, \mathbf{A}_N)^T$, $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_2, \ldots, \mathbf{B}_N)^T$, and

$$\mathbf{A}_i = (\mathbf{p}_i/m + \mathbf{i}\gamma q_{yi}, \mathbf{0}, \gamma)^T,$$
$$\mathbf{B}_i = (\mathbf{0}, \mathbf{F}_i - \mathbf{i}\gamma p_{yi} - \alpha_g\mathbf{p}_i, 0)^T.$$

The equations of motion associated with the vector field $\mathbf{A}$ are

$$\dot{\mathbf{q}}_i = \mathbf{p}_i/m + \mathbf{i}\gamma q_{yi}, \tag{11}$$

$$\dot{\mathbf{p}}_i = \mathbf{0}, \tag{12}$$

$$\dot{d}_x = \gamma, \tag{13}$$

whereas those with the vector field **B** are:

$$\dot{\mathbf{q}}_i = \mathbf{0}, \tag{14}$$

$$\dot{\mathbf{p}}_i = \mathbf{F}_i - \mathbf{i}\gamma p_{yi} - \alpha_g \mathbf{p}_i, \tag{15}$$

$$\dot{d}_x = 0. \tag{16}$$

The analytical solutions to the equations of motion (11)–(13) associated with **A** are:

$$\begin{cases} q_{xi}(\Delta t) = q_{xi}(0) + \Delta t[(1/m)p_{xi}(0) + \gamma q_{yi}(0)] \\ \qquad\qquad + (1/2m)\Delta t^2 \gamma p_{yi}(0), \\ q_{yi}(\Delta t) = q_{yi}(0) + (\Delta t/m)p_{yi}(0), \\ q_{zi}(\Delta t) = q_{zi}(0) + (\Delta t/m)p_{zi}(0), \\ d_x(\Delta t) = d_x(0) + \Delta t\gamma, \\ \mathbf{p}_i(\Delta t) = \mathbf{p}_i(0). \end{cases} \tag{17}$$

These solutions represent a change in the Cartesian coordinates of the particles while their momentum is fixed. We note here that the kinetic energy is not altered by the **A** partition of the equations of motion.

As far as we are aware, there is no *exact* analytical solution for the equations of motion associated with the vector field **B** unless the strain rate vanishes, i.e., $\gamma = 0$ (the case of the equilibrium Gaussian thermostatted equations of motion).[14] However, it has been shown that by carrying out an additional splitting of **B**, a kinetic energy conserving, reversible set of equations that can be solved analytically can be obtained.[24] This splitting and the solutions to the equations of motion are presented in Appendix B. Here we consider the numerical solution of **B**. We note that in order to apply the operator-splitting methods to construct an integrator of order r, we only need an rth order approximation of the solution (if the exact one is not available) of each split equation. Therefore we solve the equations of motion associated with the vector field **B** by considering the following finite difference scheme:

$$\frac{\mathbf{p}_i(\Delta t) - \mathbf{p}_i(0)}{\Delta t} = \mathbf{F}_i - \mathbf{i}\gamma \frac{p_{yi}(\Delta t) + p_{yi}(0)}{2} - \alpha_g \frac{\mathbf{p}_i(\Delta t) + \mathbf{p}_i(0)}{2}, \tag{18}$$

where

$$\alpha_g = \frac{\sum_{i=1}^{N}[\mathbf{F}_i \cdot (\mathbf{p}_i(\Delta t) + \mathbf{p}_i(0))/2 - \gamma(p_{xi}(\Delta t) + p_{xi}(0))(p_{yi}(\Delta t) + p_{yi}(0))/4]}{\sum_{i=1}^{N}(\mathbf{p}_i(\Delta t) + \mathbf{p}_i(0)) \cdot (\mathbf{p}_i(\Delta t) + \mathbf{p}_i(0))/4}. \tag{19}$$

Such a scheme is similar to the midpoint rule and is accurate to Δt^2. Moreover, it conserves the kinetic energy of the system: multiplying both sides of Eq. (18) by $[\mathbf{p}_i(\Delta t) + \mathbf{p}_i(0)]$, one finds,

$$\sum_{i=1}^{N} \mathbf{p}_i(\Delta t) \cdot \mathbf{p}_i(\Delta t) \equiv \sum_{i=1}^{N} \mathbf{p}_i(0) \cdot \mathbf{p}_i(0). \tag{20}$$

The nonlinear coupled equations (18) and (19) for $\mathbf{p}_i(\Delta t)$, can be solved using an iterative method. In Appendix A we present a simple and straightforward iteration procedure that converges quickly. Note that only one force field evaluation is required for the equations associated with operator **B**, because the Cartesian coordinates of all the particles remain unchanged throughout the iterative procedure. It will be shown later that solution of Eq. (18) by the iterative method does not add substantially to the computation time required for the molecular dynamics simulations. Moreover, the iteration procedure can be fully vectorized. Nevertheless, the fully analytic solution obtained by carrying out an additional splitting and described in Appendix B (Ref. 24) provide an easily programmed and fully reversible alternative. The accuracy of the results obtained using the equations with an additional splitting have not yet been carefully tested.

Given the above solutions of the split equations, we can construct various integrators for the Gaussian thermostatted SLLOD equations using operator-splitting techniques.

In this work we consider the second-order Trotter factorizations,[25] which lead to ''leapfrog'' type integrators:

$$U(\Delta t) \approx \exp\left(\frac{\Delta t}{2}\mathbf{A}\cdot\frac{\partial}{\partial \mathbf{\Gamma}}\right)\exp\left(\Delta t\mathbf{B}\cdot\frac{\partial}{\partial \mathbf{\Gamma}}\right)\exp\left(\frac{\Delta t}{2}\mathbf{A}\cdot\frac{\partial}{\partial \mathbf{\Gamma}}\right), \tag{21}$$

and

$$U(\Delta t) \approx \exp\left(\frac{\Delta t}{2}\mathbf{B}\cdot\frac{\partial}{\partial \mathbf{\Gamma}}\right)\exp\left(\Delta t\mathbf{A}\cdot\frac{\partial}{\partial \mathbf{\Gamma}}\right)\exp\left(\frac{\Delta t}{2}\mathbf{B}\cdot\frac{\partial}{\partial \mathbf{\Gamma}}\right). \tag{22}$$

We will refer to these propagators as S1 and S2, respectively.

These integrators have the following properties.

(i) Their implementations are rather straightforward. From any given values of the dynamical variables at the nth step, we can obtain their values at the $(n+1)$th step by consecutively applying each operator (with the fractional time steps).

(ii) The integrators require minimum computer memory. The values required are the updated values of **p**, **q**, and the force field vector **F**, plus memory to store the

old values (previous step of iteration) of **p** during the iterative solution process for the momentum. See Appendix A.

(iii) The integrators are kinetic energy conserving because each of the steps (associated with operators **A** and **B**) conserves the kinetic energy of the system to any desired accuracy.

(iv) Schemes (21) and (22) are invertible.

We have therefore designed second-order integrators which conserve the kinetic energy of Gaussian thermostatted dynamics to machine accuracy. Fourth-order integrators can also be constructed using symmetric composition of symmetric integrator techniques.[2,6] However, such higher-order integrators require more force computations. Optimization of the coefficients in (8) could also improve the accuracy.[6]

The new integrators are similar to the velocity-Verlet and position-Verlet methods designed for Hamiltonian dynamics,[5] and thus should have similar stability properties. In the following section we compare the new integrators with the commonly used Runge–Kutta and Gear predictor–corrector algorithms: the conservation of kinetic energy, speed, and accuracy are considered.

In order to examine the accuracy of the simulation algorithms, the pressure, p, internal energy per particle, E/N, the shear viscosity, η, and the normal stress coefficients were calculated. The shear viscosity is given by:

$$\eta = -\frac{\langle P_{xy}\rangle}{\gamma}. \tag{23}$$

Here P_{xy} is the xy component of the pressure tensor, **P**:

$$\mathbf{P}V = \sum_i^N \mathbf{p}_i\mathbf{p}_i - \frac{1}{2}\sum_{i,j}^N \mathbf{r}_{ij}\mathbf{F}_{ij}. \tag{24}$$

We define in-plane and out-of-plane viscosities, η_- and η_0 as:

$$\eta_- = -\frac{\langle P_{xx}\rangle - \langle P_{yy}\rangle}{2\gamma}, \tag{25}$$

$$\eta_0 = -\frac{\langle P_{zz}\rangle - (\langle P_{xx}\rangle + \langle P_{yy}\rangle)/2}{2\gamma}. \tag{26}$$

III. NUMERICAL TESTS

Consider 2048 WCA particles undergoing planar Couette flow with a strain rate of $\gamma = 1$. The fluid is close to the Lennard-Jones triple point at a reduced temperature and density of $T = 0.722$ and $n = 0.8442$, respectively. We will use Lennard-Jones reduced units throughout this section. Simulations were carried out using the two new integrators, S1 and S2 and the results are compared with the fourth-order Runge–Kutta (RK4), fourth-order Gear predictor–corrector (GPC4), second-order Runge–Kutta (RK2), and second-order Gear predictor–corrector (GPC2) integrators. The Gear predictor–corrector algorithms are based on the second-order equations of motion.

Two iterative procedures for solution of the finite difference equations were considered. In both cases the thermostat multiplier was updated at each iteration by:

$$\alpha_g^{(n+1)} = \frac{\Sigma_{i=1}^N [\mathbf{F}_i \cdot (\mathbf{p}_i^{(n)}(\Delta t) + \mathbf{p}_i(0))/2 - \gamma (p_{xi}^{(n)}(\Delta t) + p_{xi}(0))(p_{yi}^{(n)}(\Delta t) + p_{yi}(0))/4]}{\Sigma_{i=1}^N (\mathbf{p}_i^{(n)}(\Delta t) + \mathbf{p}_i(0)) \cdot (\mathbf{p}_i^{(n)}(\Delta t) + \mathbf{p}_i(0))/4}, \tag{27}$$

where $\mathbf{p}(0)$ is the momentum at the beginning of the timestep, and $\mathbf{p}^n(\Delta t)$ is the nth iteration value of the momentum $\mathbf{p}(\Delta t)$. In the first case (labeled It-I), the approximation for the momenta at the $(n+1)$th iteration is calculated as:

$$\mathbf{p}_i^{(n+1)}(\Delta t) = \mathbf{p}_i(0) + \Delta t \mathbf{F}_i - \mathbf{i}\gamma \Delta t \frac{p_{yi}^{(n)}(\Delta t) + p_{yi}(0)}{2} - \alpha_g^{(n)} \Delta t \frac{\mathbf{p}_i^{(n)}(\Delta t) + \mathbf{p}_i(0)}{2}, \tag{28}$$

and convergence is tested by monitoring the departure of the kinetic energy from the set value. For the second iterative scheme (labeled It-II), the $(n+1)$th iteration for the momenta is:

$$\mathbf{p}_i^{(n+1)}(\Delta t) = \left[\mathbf{p}_i(0)\left(1 - \frac{\Delta t}{2}\alpha_g^{(n)}\right) + \Delta t \mathbf{F}_i - \mathbf{i}\gamma \Delta t \frac{p_{yi}^{(n)}(\Delta t) + p_{yi}(0)}{2}\right]\left[1 + \frac{\Delta t}{2}\alpha_g^{(n)}\right]^{-1}, \tag{29}$$

and convergence is tested by monitoring the convergence of α_g to a constant value. The code for the second iterative scheme (which proved to be quicker) is given in Appendix A. In both cases convergence was fast, with the second scheme converging to within 1×10^{-9} in approximately eight iterations, depending on the timestep used. We note that there are many other iterative solutions of Eq. (18).

First, we examine the behavior of the instantaneous temperature of the system simulated using the new integrators and the conventional ordinary differential equation (ODE) solvers. For each integrator steady states were achieved with the desired temperature being maintained during the initial period using proportional feedback or rescaling. The feedback or rescaling was then switched off and a trajectory was simulated using different integrators and timestep sizes. We monitored the instantaneous temperature and observed that the RK and GPC exhibit persistent kinetic-energy drift even for very small timesteps. This drift makes it impossible to carry out simulations over long periods of time, unless an *ad hoc* method is used to maintain the temperature. In contrast, the new integrators show virtually no drift; they conserve the

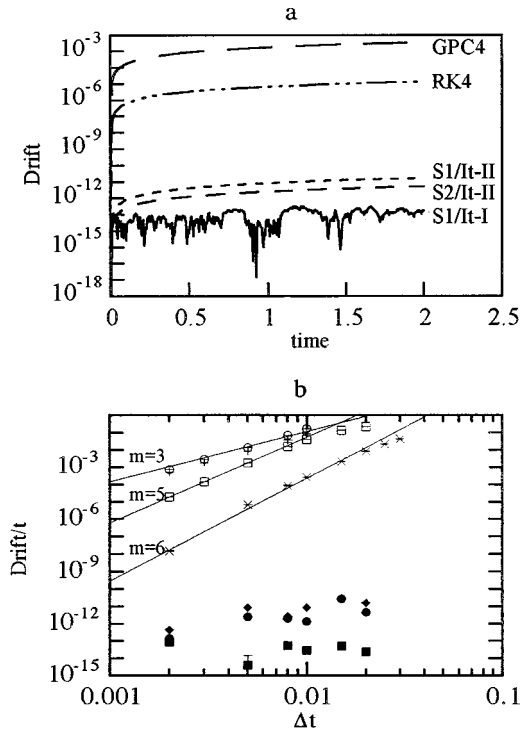

FIG. 1. (a) The behavior of the instantaneous temperature of the system using the RK4, GPC4, S1/It-I, S1/It-II, and S2/It-II integrators with a timestep of $\Delta t=0.005$. (b) The drift per unit time of the temperature as a function of timestep for various integrators: RK4 (×), GPC4 (□), RK2 (+), GPC2 (○), S1/It-I (■), S1/It-II (◆), S2/It-II (●). The straight lines of slope m are the expected slopes for these integrators. A tolerance 1×10^{-8} was used to test for the convergence when an iterative method is used to solve the equations of motion.

kinetic energy of the system to machine accuracy. The ensemble average drift is defined as,

$$\text{Drift}=\langle T(t)-T(0)\rangle. \tag{30}$$

Figure 1(a) shows the drift over a period of $t=2$ for the RK4, GPC4, S1/It-I, S1/It-II, and S2/It-II integrators when a timestep of $\Delta t=0.005$ is employed and 100 trajectories are averaged. The It-I scheme results in no drift because convergence of (28) is tested by monitoring the departure of the temperature from the desired temperature; whereas a slight drift is obtained when It-II is used because this scheme is based on the convergence of the thermostat multiplier. For the iterative methods, the slight drift in the temperature can be reduced by refining the appropriate convergence criteria. For the iteration scheme, It-II (see Appendix A), a maximum drift per unit time of 2×10^{-11} was observed for a timestep of $\Delta t=0.02$ and a tolerance of 1×10^{-8}.

In Fig. 1(b) we show the average drift per unit time of the temperature, Drift/t, for various timesteps, where the average is calculated over 100 trajectories. Again it is clear that the conventional integrators suffer from drift. The straight lines of slope m indicate the expected results based on the error estimates of the algorithms. For the new integrators, a tolerance of 1×10^{-8} was used in all iterations. This small but nonzero tolerance results in the slight drift (for S1/It-II and S2/It-II), which can be reduced or eliminated by using smaller tolerance in the test for convergence.

TABLE I. Relative CPU time taken for a simulation at $T=0.722$, $n=0.8442$, $\gamma=1.0$, and $\Delta t=0.002$. Measurements were done for simulations of length $t=20$. The times are relative to the S1 integrator with the second iterative method (It-II). The simulations were carried out on a single processor of a Silicon Graphics ORIGIN 2000.

Integrator	Relative time
Fourth-order Runge–Kutta	3.74
Fourth-order Gear predictor–corrector	1.00
Second-order Runge–Kutta	1.87
Second-order Gear predictor–corrector	0.97
Operator split S1 with It-I	1.01
Operator split S1 with It-II	1
Operator split S2 with It-II	1.06

In Table I the CPU time taken by each of the methods is compared. The time required for a simulation of 10 000 timesteps, relative to the fastest integrator developed by operator splitting (S1 with It-II) is shown. All data was generated for a timestep of $\Delta t=0.002$ and a tolerance of 1×10^{-9} for the convergence of the iterative schemes. The speeds of the integrators are consistent with the dominance of the force calculations in the time taken for the simulations, that is, the time taken is roughly proportional to the number of force calculations per step for each integrator. We see that the operator-splitting integrators require between one quarter and two sevenths (depending on the iteration scheme and convergence criterion) of the computer time used for the RK4 algorithm on the same machine. The GPC4 method is fast because it needs only one force field evaluation per step. Method S1 is slightly faster than S2 because the former uses operator **B** once per step thus involving fewer iterations. Note that relative timings may vary with machine and code details.

We investigated the largest possible timestep that could be used for each integrator, and found that S1 and S2 are stable for Δt up to approximately 0.02, RK4 to approximately 0.03, and GPC4 up to approximately 0.02.

Simulations also examined the accuracy of properties (pressure, internal energy, viscosity, and the normal stress coefficients) calculated using each of the numerical integrators as a function of the timestep. Since the Runge–Kutta and Gear predictor–corrector methods exhibit persistent kinetic energy drift, they cannot be used to obtain simulation results for long trajectories unless some additional procedure is employed to prevent the kinetic energy drift. It is a common practice to add a feedback term to the equation of motion. Feedback multipliers were first developed and employed by Baranyai and Evans.[26] Here the feedback multiplier

$$\alpha_f=(k_B\mathcal{T}-k_BT)/(Qk_BT), \tag{31}$$

where $\mathcal{T}$ is the current kinetic temperature and T is the desired value, is used to adjust the Gaussian thermostat multiplier, α_g. The magnitude of this term is controlled by the parameter, Q, which should be sufficiently large that the feedback multiplier only compensates for numerical error, and yet small enough that the temperature can be constrained. A value of $Q\approx10$ is quite suitable.

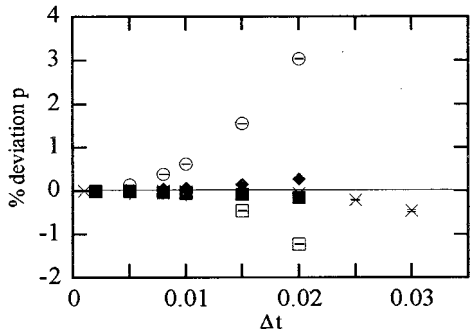

FIG. 2. The deviation of the pressure as a function of timestep calculated using simulations with various integrators: RK4 (×), GPC4 (□), RK2 (+), GPC2 (○), S1/It-II (■), S2/It-II (♦).

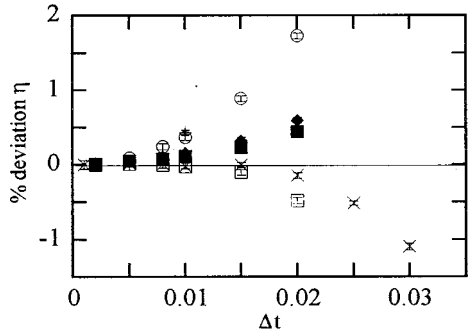

FIG. 4. The deviation of the shear dependent viscosity as a function of timestep calculated using simulations with various integrators: RK4 (×), GPC4 (□), RK2 (+), GPC2 (○), S1/It-II (■), S2/It-II (♦).

The deviations of the simulated values of the pressure, internal energy, viscosity and the normal stress coefficients from the results obtained using the most accurate simulation (fourth-order Runge–Kutta with a *very small* timestep of $\Delta t = 0.001$) are shown in Fig. 2–6. Here we use these figures to access the accuracy of the integrators for a range of properties (note that the deviations vary with the property considered). In addition we expect that these figures will provide a *useful reference* from which simulators can determine the appropriate timestep required for obtaining certain accuracy in their simulation run.

In general the results presented in Figs. 2–6 indicate that the new integrators S1 and S2 perform well: they are more stable and accurate than the other second-order methods and in most cases are more accurate than the fourth-order Gear predictor–corrector method. The fourth-order Runge–Kutta method was found to be the most stable and therefore a relatively large timestep could be used. However the RK4 method is *much slower* than the S1 or S2 methods. The S1 and S2 integrators give particularly accurate results for the pressure, total energy, and in-plane viscosity.

The statistical error in the in-plane normal stress coefficient was larger than the deviation of the result from the expected value in most cases. This high statistical error is a consequence of the in-plane normal stress coefficient being defined by the difference of two values of similar magnitude [see Eq. (25)], and in order to obtain more precise values, significantly longer simulation runs are required. At the level of precision obtained in this work, there is no advantage in using a timestep less than approximately $\Delta t = 0.02$. Therefore, due to their speed the S1, S2, and GPC4 integrators with a timestep of 0.02 would be the best methods to use for the calculation of the in-plane normal stress coefficient at this level of precision.

In Fig. 7 the CPU time required for the specified levels of accuracy of p, E/N, and η is presented; this provides information on the relative efficiencies of the integrators for each of these properties. The results were obtained by combining the data for the accuracy of the properties shown in Figs. 2–4 and the data for the simulation times presented in Table I. By interpolation, the timestep required to obtain the desired accuracy was determined from Figs. 2–4, then the simulation time was calculated from the relative speeds of the integrators. Note that the time required for higher accuracy is reduced for increasing timestep up to the stability limit of the method. The S1 method is the most efficient followed by S2 and GPC4. However, GPC4 is less accurate than the S1 and S2 methods for some properties, requires more memory, feedback, or rescaling to prevent drift and it is not self-starting.

IV. CONCLUDING REMARKS

In summary, we have developed some operator-splitting integrators for the Gaussian thermostatted SLLOD equations

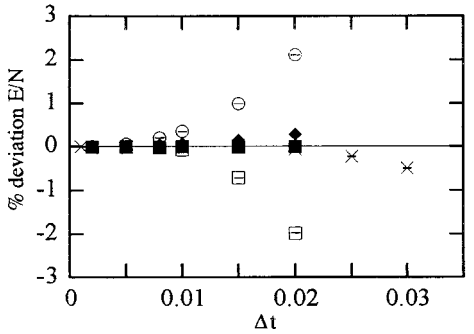

FIG. 3. The deviation of the internal energy as a function of timestep calculated using simulations with various integrators: RK4 (×), GPC4 (□), RK2 (+), GPC2 (○), S1/It-II (■), S2/It-II (♦).

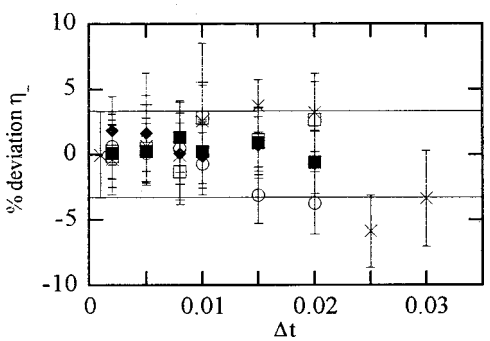

FIG. 5. The deviation of the in-plane viscosity as a function of timestep calculated using simulations with various integrators: RK4 (×), GPC4 (□), RK2 (+), GPC2 (○), S1/It-II (■), S2/It-II (♦). The lines are the error bounds for the most precise calculation carried out using conventional integrators (RK4 with a timestep of 0.001).

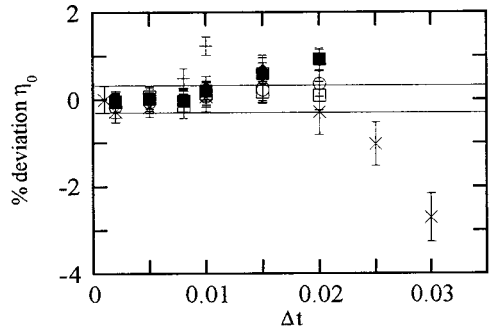

FIG. 6. The deviation of the out-of-plane viscosity as a function of timestep calculated using simulations with various integrators: RK4 (×), GPC4 (□), RK2 (+), GPC2 (○), S1/It-II (■), S2/It-II (◆). The lines are the error bounds for the most precise calculation carried out using conventional integrators (RK4 with a timestep of 0.001).

of motion. The new integrators are kinetic energy conserving, and unlike the conventional ODE solvers do not require *ad hoc* rescaling of velocities or a feedback term in the equations of motion. Although the present integration scheme is implicit, the overall speeds of the integrators are faster than the conventional integrators due to the CPU effectiveness of the iterative solution of the momenta equations. Furthermore, the integrators require minimal computer memory, and one force field evaluation per time step. Most importantly, the new integrators are stable and accurate even for large time steps. It is shown that to achieve comparable accuracy for physical properties $(p, E/N, \eta, \eta_0)$, the new integrators require much less computer time than with traditional algorithms. Therefore, the new integrators are clearly the method of choice for solution of the Gaussian thermostatted SLLOD equations of motion.

It is worth noting that a similar scheme can be applied to the isoenergetic SLLOD equations of motion. However, in this case the numerical integrator does not conserve the energy exactly at each substep of the integration since changes in the coordinates and the momenta occur simultaneously.

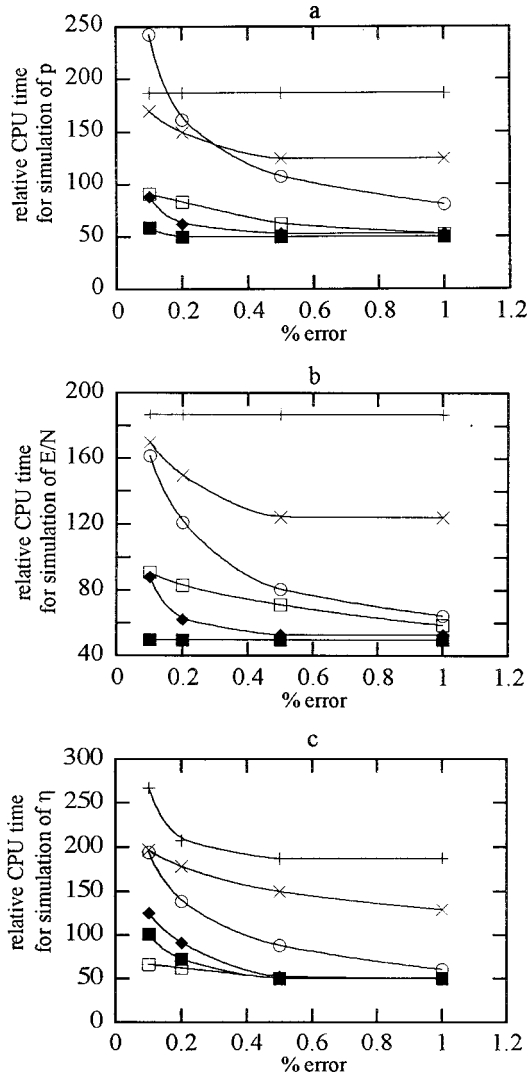

FIG. 7. The CPU time required to obtain desired levels of accuracy for various properties: (a) pressure, (b) internal energy, (c) the shear-dependent viscosity. The lines are for guidance only.

ACKNOWLEDGMENTS

Part of the work (by F.Z.) was supported by the Academic Research Grants (Nos. RP950601 and 960689) and Lee Kuan Yew Endowment Fund in the National University of Singapore. D.J.S., D.J.E., and D.J.I. acknowledge the support obtained from the Australia Research Council. We thank G. J. Martyna for his helpful comments.

APPENDIX A

```
      subroutine sllodb_itii(step, tol, px, py, pz, x, y, z, fx, fy,
fz, gamma)
   implicit none
   integer np
   parameter (np=2048)
      double precision x(np), y(np), z(np)
      double precision px(np), py(np), pz(np)
      double precision fx(np), fy(np), fz(np)
      double precision pxold(np), pyold(np), pzold(np)
      double precision step, tol, anum, aden, alph, alpha_old,
a1, a2, gamma
      integer i, it_max, it, np, maxcyc
c
      alpha_old=100000.0d0
c maximum iteration number
      it_max=20
c store momenta at the beginning of the timestep
      do 10 i=1, np
         pxold(i)=px(i)
         pyold(i)=py(i)
         pzold(i)=pz(i)
10 continue
c
      do 20 it=1, it_max
      anum=0.0d0
      aden=0.0d0
         do 30 i=1, np
      aden=aden+0.25d0*((pxold(i)+px(i))**2+
     &          (pyold(i)+py(i))**2+
     &          (pzold(i)+pz(i))**2)
```

```
      anum=anum+0.5d0*(fx(i)*(px(i)+pxold(i))+
     &                 fy(i)*p(py(i)+pyold(i))+
     &                 fz(i)*(pz(i)+pzold(i)))
     &          -0.25d0*gamma*(px(i)+pxold(i))*
     &                 (py(i)+pyold(i))
30 continue
      alph=anum/aden
      a1=1.0d0/(1.0d0+0.5d0*alph*step)
      a2=1.0d0-0.5d0*alph*step
      do 40 i=1, np
        pz(i)=a1*(step*fz(i)+a2*pzold(i))
        py(i)=a1*(step*fy(i)+a2*pyold(i))
        px(i)=a1*(step*(fx(i)-
     &      0.5d0*gamma*(py(i)+pyold(i)))+a2*pxold(i))
40 continue
c The tolerance is set by user
      if (abs(alph-alpha_old).le.tol) goto 999
      alpha_old=alph
20 continue
999 continue
      if (it.gt.maxcyc) maxcyc=it
      if (maxcyc.ge.it_max) stop 'too many iterations'
      return
      end
```

APPENDIX B

It has been shown[24] that a fully analytical, kinetic energy conserving, reversible solution to the Gauss thermostatted SLLOD equations of motion can be obtained by splitting the equations of motion associated with the vector field **B**. The propagator defined in Eq. (7) is decomposed so that $U(\Delta t) = \exp(\Delta t(\mathbf{A}+\mathbf{B_1}+\mathbf{B_2})\cdot\partial/\partial\mathbf{\Gamma})$ where $\mathbf{B_1}$ is

$$\dot{\mathbf{q}}_i = \mathbf{0}, \tag{B1}$$

$$\dot{\mathbf{p}}_i = -\mathbf{i}\gamma p_{yi} - \alpha_{g1}\mathbf{p}_i, \tag{B2}$$

$$\dot{d}_x = 0, \tag{B3}$$

$\mathbf{B_2}$ is

$$\dot{\mathbf{q}}_i = \mathbf{0}, \tag{B4}$$

$$\dot{\mathbf{p}}_i = \mathbf{F}_i - \alpha_{g2}\mathbf{p}_i, \tag{B5}$$

$$\dot{d}_x = 0, \tag{B6}$$

and where

$$\alpha_{g1} = \frac{1}{2mK_0}\sum_{j=1}^{N}[-\gamma p_{xj}p_{yj}], \tag{B7}$$

$$\alpha_{g2} = \frac{1}{2mK_0}\sum_{j=1}^{N}[\mathbf{F}_j\cdot\mathbf{p}_j]. \tag{B8}$$

The solution to the equations associated with vector field $\mathbf{B}_1$ are:[24]

$$\mathbf{q}_i(\Delta t) = \mathbf{q}_i(0), \tag{B9}$$

$$\mathbf{p}_i(\Delta t) = g(\Delta t)(\mathbf{p}_i(0) - \mathbf{i}\Delta t\,\gamma p_{yi}), \tag{B10}$$

$$d_x(\Delta t) = d_x(0), \tag{B11}$$

where

$$g(\Delta t) = (1 - 2C_1\Delta t + C_2\Delta t^2)^{-1/2}, \tag{B12}$$

$$C_1 = \frac{\gamma}{2mK_0}\sum_{j=1}^{N}[p_{xj}(0)p_{yj}(0)], \tag{B13}$$

$$C_2 = \frac{\gamma^2}{2mK_0}\sum_{j=1}^{N}[p_{yj}(0)p_{yj}(0)]. \tag{B14}$$

Those associated with vector field $\mathbf{B_2}$ are simply those of the field free system with solutions:[14]

$$\mathbf{q}_i(\Delta t) = \mathbf{q}_i(0), \tag{B15}$$

$$\mathbf{p}_i(\Delta t) = \frac{1-h}{f(\Delta t)-h/f(\Delta t)}\left[\mathbf{p}_i(0) + \mathbf{F}_i(0)\,\frac{1+h-f(\Delta t)-h/f(\Delta t)}{(1-h)\beta}\right], \tag{B16}$$

$$d_x(\Delta t) = d_x(0), \tag{B17}$$

where

$$\alpha(\Delta t) = \beta\left[\frac{\alpha(0)+\beta\tanh(\beta\Delta t)}{\beta+\alpha(0)\tanh(\beta\Delta t)}\right], \tag{B18}$$

$$\beta = \left[\frac{1}{2mK_0}\sum_{j=1}^{N}(\mathbf{F}_j\cdot\mathbf{F}_j)\right]^{1/2}, \tag{B19}$$

$$f(\Delta t) = \exp(-\beta\Delta t), \tag{B20}$$

$$h = \frac{\alpha(0)+\beta}{\alpha(0)-\beta}. \tag{B21}$$

[1] M. Suzuki, J. Math. Phys. **26**, 601 (1985); Phys. Lett. A **165**, 387 (1992); J. Phys. Soc. Jpn. **61**, 3015 (1992).
[2] H. Yoshida, Phys. Lett. A **150**, 262 (1990); Celest. Mech. Dyn. Astron. **56**, 27 (1993).
[3] E. Forest and R. D. Ruth, Physica D **43**, 105 (1990).
[4] J. Candy and W. Rozmus, J. Comput. Phys. **92**, 230 (1991).
[5] M. E. Tuckerman, J. Berne, and G. J. Martyna, J. Chem. Phys. **97**, 1990 (1992).
[6] R. I. McLachlan and P. Atela, Nonlinearity **5**, 541 (1992); R. I. McLachlan, SIAM (Soc. Ind. Appl. Math.) J. Sci. Stat. Comput. **16**, 151 (1995).
[7] D. I. Okunbor and R. D. Skeel, J. Comput. Chem. **15**, 72 (1994); S. K. Gray, D. W. Noid, and B. G. Sumpter, J. Chem. Phys. **101**, 4062 (1994).
[8] G. J. Martyna and M. E. Tuckerman, J. Chem. Phys. **102**, 8071 (1995).
[9] G. Benettin and A. Giogilli, J. Stat. Phys. **74**, 1117 (1994).
[10] Y. F. Tang, Comput. Math. Appl. **27**, 31 (1994).
[11] G. J. Martyna, M. E. Tuckerman, D. J. Tobias, and M. L. Klein, Mol. Phys. **87**, 1117 (1996).
[12] D. J. Evans and G. P. Morriss, *Statistical Mechanics of Nonequilibrium Liquids* (Academic, London, 1990).
[13] M. E. Tuckerman and M. Parrinello, J. Chem. Phys. **101**, 1302 (1994).
[14] F. Zhang, J. Chem. Phys. **106**, 6102 (1997).
[15] Z. Xu, J. J. de Pablo, and S. Kim, J. Chem. Phys. **102**, 5836 (1995).
[16] C. J. Mundy, J. I. Siepmann, and M. L. Klein, J. Chem. Phys. **103**, 10192 (1995).
[17] S. T. Cui, P. T. Cummings, and H. D. Cochran, J. Chem. Phys. **104**, 255 (1996).
[18] D. J. Isbister, D. J. Searles, and D. J. Evans, Physica A **240**, 105 (1997).
[19] J. Petravic and D. J. Evans, Mol. Phys. **95**, 219 (1998).

[20] H. J. C. Berendsen and W. F. van Gunsteren, *Proceedings of the Enrico Fermi Summer School, Vareena, 1985* (Soc. Italiana di Fisica, Bologna), p. 43.
[21] J. S. dT. Hansen, D. J. Isbister, D. J. Searles, F. Zhang, and D. J. Evans (unpublished).
[22] Note that the SLLOD equations of motion are nonautonomous so the Jacobian has an explicit time-dependence. This time-dependence can be treated by introducing an additional parameter, the lattice strain and considering an extended phase space Jacobian, but its effects on phase variables is negligible for system sizes greater than $N=8$. See Ref. 19.
[23] D. J. Evans, Phys. Rev. A **34**, 1449 (1986).
[24] G. J. Martyna (private communication).
[25] H. F. Trotter, Proc. Am. Math. Soc. **10**, 545 (1959).
[26] A. Baranyai and D. J. Evans, Mol. Phys. **70**, 53 (1990).